# A Systematic Study of Maximal Scheduling Algorithms in Multiradio Multichannel Wireless Networks

Yu Cheng, *Senior Member, IEEE*, Hongkun Li, *Member, IEEE*, Devu Manikantan Shila, and Xianghui Cao, *Member, IEEE*

***Abstract*—The greedy maximal scheduling (GMS) and maximal scheduling (MS) algorithms are well-known low-complexity scheduling policies with guaranteed capacity region in the context of single-radio single-channel (SR-SC) wireless networks. However, how to design maximal scheduling algorithms for multiradio multichannel (MR-MC) wireless networks and the associated capacity analysis are not well understood yet. In this paper, we develop a new model by transforming an MR-MC network node to multiple node-radio-channel (NRC) tuples. Such a framework facilitates the derivation of a tuple-based back-pressure algorithm for throughput-optimal control in MR-MC wireless networks and enables the tuple-based GMS and MS scheduling as low-complexity approximation algorithms with guaranteed performance. An important existing work on GMS and MS for MR-MC networks is that of Lin and Rasool (*IEEE/ACM Trans. Networking*, vol. 17, no. 6, 1874–1887, Dec. 2009), where link-based algorithms are developed. Compared to the link-based algorithms, the tuple-based modeling has significant advantages in enabling a fully decomposable cross-layer control framework. Another theoretical contribution in this paper is that we, for the first time, extend the local-pooling factor analysis to study the capacity efficiency ratio of the tuple-based GMS in MR-MC networks and obtain a lower bound that is much tighter than those known in the literature. Moreover, we analyze the communications and computation overhead in implementing the distributed MS algorithm and present simulation results to demonstrate the performance of the tuple-based maximal scheduling algorithms.**

***Index Terms*—Capacity region, local-pooling factor, maximal scheduling, multiradio multichannel, throughput-optimal control.**

## I. Introduction

THE MULTIRADIO multichannel (MR-MC) networking can significantly increase network capacity by simultaneously exploiting multiple nonoverlapping channels through different radio interfaces and mitigating interferences through proper channel assignment [1]–[3]. Compared to the traditional single-radio single-channel (SR-SC) networks, MR-MC networking takes place in a *multidimensional resource space*, with dimensions defined by radio interfaces, links, and channels. In this paper, we give a systematic study of throughput-optimal control algorithms in such a multidimensional resource space, with particular focus on low-complexity implementations with guaranteed capacity region.

The *throughput-optimal* schedule has been extensively studied in the SR-SC context [4]–[6], which aims at maintaining the network stability as long as the traffic dynamics are within the capacity region. The central component of the throughput-optimal algorithms is the *back-pressure-based scheduling* [4], [7], which can be reduced to a maximum weighted independent set (MWIS) problem and is NP-hard in general. The *greedy maximal scheduling* (GMS) [5], [8] is a well-known low-complexity algorithm that can approximate the throughput-optimal control with guaranteed performance. The performance of a scheduling policy is normally characterized by the *capacity efficiency ratio*, defined as the largest achievable fraction of the optimal capacity region [10], [11]. There are some studies on the capacity efficiency ratio of the GMS algorithm [9], [11]–[13]; the tightest ratio is derived in [11] through a local-pooling factor analysis. While the performance of GMS is often close to optimal, it is a centralized algorithm. The *maximal scheduling* (MS) [5], [10], [12] is a popular distributed algorithm with guaranteed capacity region.

In the MR-MC wireless networks, the scheduling involves a set of coupled resource allocation problems including link scheduling, channel/radio assignment, and routing [1], [14], [16], [17]. How to achieve throughput-optimal control in such a link-radio-channel multidimensional resource space has not been systematically studied. The current state of the art still stands on link-based back-pressure formulation, while it resorts to heuristic algorithms to make decisions on channel/radio assignment [16], [17]. Lin and Rasool generalize the GMS and MS algorithms to the MR-MC context in [16]. In the GMS algorithm there, a common link queue is used to determine the scheduling over different channels; we will show that such an implementation cannot lead to the best performance in cross-layer control. Furthermore, the capacity efficiency ratio of GMS in [16] is based on the *interference degree*, which has been shown to be a loose bound [11]. The local-pooling-factor-based analysis [11], which can generate a tight lower bound of the GMS capacity efficiency ratio, has not been studied yet in the

Manuscript received June 21, 2012; revised April 20, 2013; October 19, 2013; and March 21, 2014; accepted May 10, 2014; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor S. Weber. Date of publication June 02, 2014; date of current version August 14, 2015. This work was supported in part by the NSF under Grant CNS-1320736 and CAREER Award Grant CNS-1053777.

Y. Cheng and X. Cao are with the Department of Electrical and Computer Engineering, Illinois Institute of Technology, Chicago, IL 60616 USA (e-mail: cheng@iit.edu; xcao10@iit.edu).

H. Li is with InterDigital, Inc., King of Prussia, PA 19406 USA (e-mail: Hongkun.Li@InterDigital.com).

D. M. Shila is with the United Technologies Research Center, Hartford, CT 06108 USA (e-mail: devums@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2014.2324976

MR-MC context, but is to be conducted in this paper. Moreover, we will present a systematical approach to streamline and simplify the design and capacity analysis for a distributed MS algorithm for the MR-MC networks.

It has been shown that the throughput-optimal control can be viewed as a dynamic implementation of a subgradient search method to solve the multicommodity flow (MCF) problem using convex duality [15], [17]. In the SR-SC context, the dual formulation allows integrating the scheduling with different optimization objective functions to form a cross-layer network control framework [15]–[17], [20]. In the MR-MC context, a fully decomposable cross-layer framework cannot be achieved with the existing link-based scheduling formulations, which are still coupled with the channel and radio assignment issues. The cross-layer control in [16] just heuristically combines path selection with the maximal scheduling to achieve a portion of the network capacity. The cross-layer framework developed in [17] relies on the algorithms in [16] to deal with the issues of link scheduling and radio and channel assignment. Alicherry *et al.* [1] study the joint channel assignment and routing for optimal throughput in MR-MC networks, but in the context of static network planning.

In this paper, we systematically study the throughput-optimal control in MR-MC networks by developing a new modeling framework. Specifically, we transform a network node to multiple *node-radio-channel (NRC) tuples*; each communication link is then mapped to multiple *tuple links*. With the tuple-based formulation, we can rigorously derive a decomposable, cross-layer optimization framework that consists of congestion control, traffic distribution over tuples, and throughput-optimal scheduling. In particular, the throughput-optimal scheduling is in the form of a *tuple-based back-pressure* algorithm. In the multidimensional resource space, the optimal scheduling of tuples can jointly indicate the optimal solutions of link scheduling, channel assignment, and radio assignment by mapping the dimensions of the scheduled tuples to resource allocation decisions. For the case of cross-layer control with path selection, we will show that the tuple-based model enables channel-dependent path selection and thus leads to a larger capacity region, compared to that in [16].

The tuple-based model in fact provides a virtual SR-SC network equivalent to the original MR-MC network, which readily allows a clean formulation of the GMS and MS algorithms based on tuple links. Such a clean formulation then enables us to extend the local-pooling factor analysis [11] to study the capacity efficiency ratio of the tuple-based GMS and obtain a lower bound of the ratio that is much tighter than the lower bound in [16]. We also analyze the message exchange communication overhead when the tuple-based MS is implemented with a randomized distributed algorithm [19], [23]. We theoretically show that the tuple-based MS provisions a mechanism to mitigate the impact of communication overhead on the throughput by properly selecting the transmission period within each scheduling slot.

The remainder of this paper is organized as follows. Section II reviews the link-based GMS and MS algorithms. Section III presents the tuple-based network model. Section IV derives a decomposable, cross-layer optimization framework and formulates the tuple-based throughput-optimal control. Sections V and VI study the tuple-based GMS and MS algorithms and their capacity regions, respectively. Section VII presents the simulation results. Section VIII reviews more related work. Section IX gives the concluding remarks.

## II. Link-Based Algorithms

In order to better demonstrate the new contributions of this paper through comparison, we here summarize the link-based GMS and MS algorithms developed in [16]. Some important concepts and terms on interference model and scheduling are also defined in this section.

Consider the wireless network as a directed graph $G(\mathcal{N}, \mathcal{L})$ with node set $\mathcal{N}$ and link set $\mathcal{L}$. Let $b(l)$ and $e(l)$ denote the transmitter node and receiver node of link $l$, respectively. A node $n$ is equipped with $M_n$ radio interfaces. Let $E(n)$ denote the set of all links originating or terminating at node $n$. The whole spectrum available to the network is divided into $C$ frequency channels. Considering the channel diversity, we use $w_l^c$ to denote the capacity of link $l$ on channel $c$. At any given time, an interface can only tune to one channel, but it can switch its channel dynamically at different time.

The interference is defined according to the *protocol interference model* [22], where a link could have a successful transmission if there is no other simultaneous transmissions within the *interference range*. Assume all the links have the same interference range over all channels. If the ratio between the interference range and the communication range is $K$, the interference model can also be termed as a *$K$-hop interference model* [23]. For each link $l$, let $I(l)$ denote the set of links that interfere with $l$ over the same channel. For convenience, we adopt the convention that $l \in I(l)$. The *interference degree* $\mathcal{K}(l)$ is defined as the maximum number of links in $I(l)$ that do not interfere with each other, when they work on the same channel. The interference degree $\mathcal{K}$ of the whole network is the maximum interference degree over all links.

We consider a *slotted* system, where time is divided into slots of unit length. A set of active links selected in a time-slot form a *feasible schedule* if none of them interferes with each other. In an MR-MC wireless network, the transmission over a link $l$ may include parallel transmissions over separate radio interfaces on different channels [16]. Thus, the link scheduling directly depends on the radio and channel assignment. For ease of exposition, in the rest of the paper whenever there is no source of confusion, we use the term "schedule" to refer to all the resource allocation decisions in a time-slot including link selection and radio/channel assignment to the selected links. At time-slot $t$, let $\mathcal{M}(t) = \{\mathcal{M}_c(t)\}$ denote the outcome of the scheduling algorithm, where $\{\mathcal{M}_c(t)\}$ is the set of noninterfering links that are chosen to transmit on channel $c$ at time $t$. Let $D_l(t)$ denote the number of packets that link $l$ can serve at time-slot $t$. Then, $D_l(t) = \sum_{c:l\in\mathcal{M}_c(t)} w_l^c$.

Let $\vec{r} = [r_1, r_2, \ldots, r_{|\mathcal{L}|}]$ denote the vector of link flow rates loaded to the network. The *capacity region* under a particular scheduling algorithm is the set of $\vec{r}$ such that the system remains stable. The *optimal capacity region* $\Omega$ is defined as the supremum of the capacity regions of all algorithms. An algorithm is *throughput-optimal* if it can achieve the optimal capacity region $\Omega$. The *capacity efficiency ratio* of a suboptimal scheduling algorithm is the largest number $\gamma$ such that this algorithm can stabilize the network under any load $\vec{r} \in \gamma\Omega$.

Let $Q_{b(l)}$ and $Q_{e(l)}$ denote the queue length at the transmitter node $b(l)$ and the receiver node $e(l)$ of link $l$, respectively. The

backlog or queue length of link $l$ at time $t$ can be defined as $Q_l(t) = Q_{b(l)}(t) - Q_{e(l)}(t)$. With $D_l(t)$ denoting the link capacity in an MR-MC wireless network, the optimal capacity region can be attained by the throughput-optimal scheduling $\mathcal{M}^*(t)$ [4], which maximizes the sum of the queue-weighted link capacity. That is

$$\mathcal{M}^*(t) = \arg\max_{\mathcal{M}(t)} \sum_{l:l\in\mathcal{M}(t)} Q_l(t)D_l(t) \quad (1)$$

$$= \arg\max_{\{\mathcal{M}_c(t)\}} \sum_{(l,c):l\in\mathcal{M}_c(t)} Q_l(t)w_l^c. \quad (2)$$

The throughput-optimal scheduling is NP-hard in general [4], [7]. The reason we term the scheduling algorithms in [16] as *link-based* algorithms is that the algorithms there essentially give optimal or suboptimal solutions to (1), where the multichannel transmissions over a link is considered as a specific implementation to achieve a link capacity $D_l(t) = \sum_{c:l\in\mathcal{M}_c(t)} w_l^c$.

Note that when multiple commodity flows are considered, with each commodity indicated by a source–destination pair, the per-flow queues need to be maintained at each node. Let $Q_{b(l)}^f$ and $Q_{e(l)}^f$ denote the queue lengths associated with commodity flow $f$ at the transmitter node $b(l)$ and the receiver node $e(l)$ of link $l$, respectively. The backlog of flow-$f$ queue over link $l$ at time $t$ can be defined as $Q_l^f(t) = Q_{b(l)}^f(t) - Q_{e(l)}^f(t)$. It has been shown [15] that the throughput-optimal control in the multicommodity case is as follows: First, select flow $f^*$ at each link $l$ as $f^* = \text{argmax}_f\{Q_{b(l)}^f(t) - Q_{e(l)}^f(t)\}$, and then use $Q_l^{f^*}$ as link weight to calculate the throughput-optimal scheduling. In the remainder of this section, we omit the superscript "$f$" for discussions in a general context.

### A. Link-Based Greedy Maximal Scheduling

The greedy maximal scheduling algorithm can be viewed as an approximation to the throughput-optimal algorithm. A schedule $\mathcal{M}(t)$ is *maximal* if $\mathcal{M}(t)$ is a noninterfering schedule, and no more links can be added to $\mathcal{M}_c(t)$ at any channel $c$ without violating the interference constraint and radio interface constraint. The GMS algorithm at time-slot $t$ proposed in [16] works as follows.

- *Step 1:* Form a set $\mathcal{F}$ of all link–channel pairs $(l, c)$. Define the weight of each link–channel pair $(l, c)$ to be $Q_l(t)w_l^c$. Start from an empty schedule $\mathcal{M}(t)$.
- *Step 2:* Search for the link–channel pair $(l, c)$ with the largest weight $Q_l(t)w_l^c$. Add link $l$ to $\mathcal{M}_c(t)$. Remove from $\mathcal{F}$ all link–channel pairs $(k, c)$ with $k \in I(l)$. Furthermore, if by scheduling link $l$ on channel $c$, the transmitting node $b(l)$ (or respectively, the receiving node $e(l)$) already uses up all $M_{b(l)}$ (or respectively, $M_{e(l)}$) radio interfaces, remove from $\mathcal{F}$ all link–channel pairs $(k, c')$ with $k \in E(b(l))$ (or respectively, $k \in E(e(l))$).
- *Step 3:* Redenote the remaining pairs in $\mathcal{F}$ as a new set $\mathcal{F}$. If $\mathcal{F}$ is not empty, repeat Step 2; otherwise, stop the algorithm, and the schedule at slot $t$ has been obtained.

*Remark 1:* The link-based GMS algorithm makes the scheduling decision over all the link–channel pairs associated with link $l$ based on a common queue $Q_l(t)$. In the tuple-based GMS, the scheduling decision will be made based on decomposed tuple-link queues. It is shown in [16] that the link-based GMS can guarantee a capacity efficiency ratio of $\frac{1}{\mathcal{K}+2}$. In Section V-B, a tighter efficiency ratio will be derived for the tuple-based GMS.

### B. Link-Based Maximal Scheduling

A particular issue for the maximal scheduling in the MR-MC context is due to the channel diversity: The MS algorithm could pick the weakest links (with the least capacity) at each channel into the maximal schedule [16]. The MS algorithm in [16] introduces a *two-stage queueing* mechanism to address the channel diversity issue. The implementation details are as follows. Each link maintains $C + 1$ queues. There is one queue $Q_l$ for each link $l$, which represents the backlog of packets at link $l$ that have not been assigned to channel queues yet. At the same time, each link $l$ maintains $C$ channel queues $\eta_l^c, c = 1, \ldots, C$. The per-channel queue $\eta_l^c$ represents the backlog of packets assigned to channel $c$ by link $l$ that are still waiting to be served.

*Stage 1:* Let $x_l^c(t)$ denote the number of packets that link $l$ can assign to channel $c$ at time-slot $t$, which is determined as

$$x_l^c(t) = w_l^c, \quad \text{if } \frac{Q_l(t)}{\alpha_l} \geq \frac{1}{w_l^c}\left\{\sum_{k\in I(l)} \frac{\eta_k^c(t)}{w_k^c} + \frac{1}{M_{b(l)}} \sum_{k\in E(b(l))} \sum_{d=1}^{C} \frac{\eta_k^d(t)}{w_k^d} + \frac{1}{M_{e(l)}} \sum_{k\in E(e(l))} \sum_{d=1}^{C} \frac{\eta_k^d(t)}{w_k^d}\right\}$$

$$x_l^c(t) = 0, \qquad \text{otherwise} \quad (3)$$

where $\alpha_l$ is a positive control parameter chosen for link $l$. Depending on the amount of backlog available in the queue $Q_l$, the actual number of packets assigned to queue $\eta_l^c$ will be $y_l^c(t) \in [0, x_l^c(t)]$.

*Stage 2:* Based on the channel queues $\eta_l^c(t)$, the *multichannel maximal scheduling* is carried out to determine the channel assignment and link schedules, i.e., the scheduling of link–channel pairs. A link–channel pair $(l, c)$ is defined as backlogged if $\eta_l^c(t) \geq w_l^c$. According to the multichannel maximal scheduling, at least one of the following is true.

- Link $l$ is scheduled on channel $c$, i.e., $l \in \mathcal{M}_c(t)$.
- One of the interfering links to link $l$ is backlogged and scheduled on channel $c$, i.e., $k \in \mathcal{M}_c(t)$ for some $k \in I(l)$.
- Either the transmitter or the receiver of link $l$ has used up all the radios.

It has been proved in [16] that the two-stage MS algorithm can achieve a capacity efficiency ratio of $\frac{1}{\mathcal{K}+2}$. In [16], the algorithm is named as SP algorithm, when it is used with a single path fixed for each commodity flow; the algorithm is named as MP algorithm when it is used in a cross-layer control framework with multiple available paths.

*Remark 2:* When implemented as an MP algorithm with cross-layer control, the two-stage queue structure may not lead to the best performance. In [16], the first-stage queue length $Q_l$ is presented to network layer as a congestion indicator to facilitate path selection. However, such congestion information is not accurate, as the backlog in second-stage link–channel pair queues is not included. We will show that with the tuple-based scheduling, the queue length of a tuple link presented to the network layer can exactly indicate the backlog of a link under a certain channel and radio assignment. Simulation results will be presented to demonstrate the capacity gain in a cross-layer control with path selection, when the tuple-based MS is used instead of the two-stage MS.

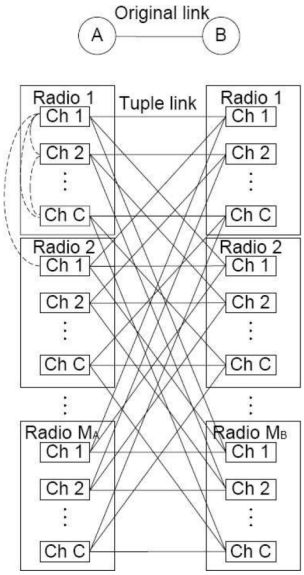



Fig. 1. Generating the tuple links.

## III. Tuple-Based Network Model

Consider the original network $G$ with node set $\mathcal{N}$ and link set $\mathcal{L}$. Each node $n \in \mathcal{N}$ is equipped with $M_n$ radio interfaces. There are $C$ channels available.

### A. Transforming to a Tuple-Based Network

We construct the tuple-based network model by transforming a node $n$ to multiple *NRC tuples*, denoted as $(n, m_n, c)$, with $m_n = 1, \ldots, M_n$ and $c = 1, \ldots, C$. The tuple $(n, m_n, c)$ means that the $m_n$th radio of node $n$ is working on channel $c$. Therefore, node $n$ is mapped to $M_n C$ tuples. We also use $p$ to denote a tuple and $n(p)$ to denote the node generating the tuple $p$. Two tuples $p$ and $p'$ form a *tuple link* if they are on the same channel, and the nodes $n(p)$ and $n(p')$ form an original link in $G$. According to such transformation rules, each original link $(i, j) \in \mathcal{L}$ can spawn $M_i M_j C$ tuple links, with each tuple link in the form of $((i, m_i, c), (j, m_j, c))$.

Let $\mathcal{P}$ and $\mathcal{L}_{\mathcal{P}}$ denote the whole tuple set and the whole tuple link set, respectively. For the convenience of presentation, we use $\ell$ or $(p, p')$ to denote a tuple link. Let $l(\ell)$ or $l(p, p')$ denote the original link that spawns the tuple link, and $c(\ell)$ or $c(p, p')$ denote the channel associated with the tuple link. The physical capacity of a tuple link $\ell$ is set the same as that of the original link $l(\ell)$. We use $w_\ell^{c(\ell)}$ to indicate the capacity of a tuple link $\ell$ and $w_\ell^{c(\ell)} = w_{l(\ell)}^{c(\ell)}$, or in the equivalent form $w_{(p,p')}^{c(p,p')} = w_{l(p,p')}^{c(p,p')}$. Fig. 1 illustrates how to transform an original link to a set of tuple links. Note that in an MR-MC network, a packet may enter and then leave a node through different radio–channel combinations. Such a behavior appears in the tuple-based network as a packet can enter and leave a network node via different tuples through *in-node transitions*. In Fig. 1, we use dotted lines to indicate the traffic transition within a node and solid lines to indicate the tuple links.

### B. Interference Model for Tuple Links

The protocol interference model [22] can be extended to the tuple-based network, considering that in MR-MC networks both co-channel contention and radio interface competence will result in an interference or conflict relationship. Specifically, two tuple links $(p_1, p_2)$ and $(p_3, p_4)$ may interfere with each other according to two types of conflict relationships.

- *Radio conflict*: Any pair of tuples—as $p_1$ and $p_3$, $p_1$ and $p_4$, $p_2$ and $p_3$, or $p_2$ and $p_4$—shares a common radio interface.
- *Co-channel interference conflict*: The original links $l(p_1, p_2)$ and $l(p_3, p_4)$ are within each other's interference range, and $c(p_1, p_2) = c(p_3, p_4)$.

With the clear definition of the interference model for the tuple links, we can see that the tuple-based network is in fact a *virtual SR-SC network model* equivalent to the original MR-MC network. While the entities for scheduling in a real SR-SC network are links, the entities for scheduling in a virtual SR-SC network are tuple links. According to the conventions in the SR-SC context, we can define $I(\ell)$ as the set of tuple links that conflict with the tuple link $\ell$ and set $\ell \in I(\ell)$. We can further define the interference degree $\mathcal{Y}(\ell)$ over the set $I(\ell)$ and the network interference degree $\mathcal{Y}$ over the whole tuple network; here, we use the notation $\mathcal{Y}$ instead of $\mathcal{K}$ to highlight that the values apply to the tuple-based network (i.e., the virtual SR-SC network).

## IV. Tuple-Based Throughput-Optimal Control

### A. Problem Formulation

We study the tuple-based throughput-optimal control within the framework of multicommodity flow optimization. Consider $F$ commodity flows over the network with the input rate vector $\vec{\lambda} = (\lambda^1, \ldots, \lambda^f, \ldots \lambda^F)$. For convenience, we use $F$ to denote the set of all the flows when the context is clear. Let $s(f)$ and $d(f)$ denote the source and destination nodes of flow $f$, respectively. Let $p_s$ denote a tuple associated with a source node $s$, and $\lambda_{p_s}^f$ denote the input rate of flow $f$ from a source tuple $p_s$. We thus have $\lambda^f = \sum_{p_s : n(p_s) = s(f)} \lambda_{p_s}^f$. Furthermore, let $\mu_{p,p'}^f$ denote the in-node transition rate of flow $f$ from tuple $p$ to $p'$, within the node $n(p) = n(p')$. The utility function of each commodity flow is $U(\lambda^f)$. The problem is to determine the flow allocation over each tuple link for each commodity flow $f$, denoted as $r_\ell^f$ or $r_{(p,p')}^f$, to maximize the total utility over the network. The utility function is assumed to be strictly concave, so our study is in the context of convex optimization.

We formulate the following convex optimization problem (**P1**) to solve the joint resource allocation:

$$\max_{\vec{\lambda}, \vec{\lambda}_{p_s}, \vec{r}, \vec{\mu}} \sum_f U(\lambda^f) \tag{4}$$

subject to:

$$\sum_{p' : (p', p) \in \mathcal{L}_{\mathcal{P}}} r_{(p', p)}^f + \sum_{p' : n(p') = n(p)} \mu_{p', p}^f \le \sum_{p'' : (p, p'') \in \mathcal{L}_{\mathcal{P}}} r_{(p, p'')}^f + \sum_{p'' : n(p'') = n(p)} \mu_{p, p''}^f \qquad \forall f \in F,\ \forall p \notin \{p_s\} \tag{5}$$

$$\sum_{p' : (p', p_s) \in \mathcal{L}_{\mathcal{P}}} r_{(p', p_s)}^f + \sum_{p' : n(p') = n(p_s)} \mu_{p', p_s}^f + \lambda_{p_s}^f \le \sum_{p'' : (p_s, p'') \in \mathcal{L}_{\mathcal{P}}} r_{(p_s, p'')}^f + \sum_{p'' : n(p'') = n(p_s)} \mu_{p_s, p''}^f \qquad \forall f \in F,\ \forall p_s : n(p_s) = s(f) \tag{6}$$

$$\lambda^f = \sum_{p_s : n(p_s) = s(f)} \lambda_{p_s}^f \qquad \forall f \in F \tag{7}$$

$$r_{(p,p')}^f \ge 0 \qquad \forall f \in F,\ \forall (p, p') \in \mathcal{L}_{\mathcal{P}} \tag{8}$$

$$\mu_{p,p'}^f \ge 0 \qquad \forall f \in F,\ \forall p \text{ and } p' \text{ with } n(p) = n(p') \tag{9}$$

$$\sum_{p':n(p')=n(p)} \sum_f \mu^f_{p',p} \le \max_{p'':(p,p'')\in\mathcal{L}_\mathcal{P}} w^{c(p,p'')}_{(p,p'')} \qquad \forall p \in \mathcal{P} \tag{10}$$

$$\vec{r} \in Co(\mathcal{M}_\mathcal{P}) \tag{11}$$

where $\vec{\lambda}$, $\vec{\lambda}_{p_s}$, $\vec{r}$, and $\vec{\mu}$ are the vectors for the optimization variables of input flow rates, input flow rates over source tuples, flow allocation over tuple links, and in-node flow transition rates, respectively.

The constraints (5) and (6) ensure flow conservation at non-source tuples and source tuples, respectively. Constraint (7) describes the distribution of input traffic over source tuples at each source node of a flow. Conditions (8) and (9) specify non-negative capacity. Condition (10) imposes a constraint on the in-node transition rates. The in-node transition rates should be large enough not to negatively impact the capacity region of the original network. We thus set an upper bound with the understanding that the total in-node transition rate to a tuple should not exceed the maximum possible output capacity for that tuple. Constraint (11) indicates that the flow allocation over the tuple-based network should be within the capacity region. It is known that the capacity region of a wireless network is defined by the convex hull of the set of maximal schedules [11]. We here use $\mathcal{M}_\mathcal{P}$ to indicate the set of tuple-link-based maximal schedules.

### B. Dual Decomposition

We introduce $Q^f_p$, $Q^f_{p_s}$, and $q^f$ as Lagrange multipliers associated with constraints (5), (6), and (7), respectively. Then, we can derive the partial Lagrange dual function as follows:

$$\begin{aligned} L(Q,q) = \max_{\vec{\lambda},\vec{r},\vec{\lambda}_{p_s},\vec{\mu}} \Bigg\{ & \sum_f U(\lambda^f) + \sum_f q^f \left( \sum_{p_s} \lambda^f_{p_s} - \lambda^f \right) \\ & + \sum_{p,f} Q^f_p \left( \sum_{p'':(p,p'')\in\mathcal{L}_\mathcal{P}} r^f_{(p,p'')} + \sum_{p'':n(p'')=n(p)} \mu^f_{p,p''} \right. \\ & \left. - \sum_{p':(p',p)\in\mathcal{L}_\mathcal{P}} r^f_{(p',p)} - \sum_{p':n(p')=n(p)} \mu^f_{p',p} \right) \\ & + \sum_{p_s,f} Q^f_{p_s} \left( \sum_{p'':(p_s,p'')\in\mathcal{L}_\mathcal{P}} r^f_{(p_s,p'')} + \sum_{p'':n(p'')=n(p_s)} \mu^f_{p_s,p''} \right. \\ & \left. \left. - \sum_{p':(p',p_s)\in\mathcal{L}_\mathcal{P}} r^f_{(p',p_s)} - \sum_{p':n(p')=n(p_s)} \mu^f_{p',p_s} - \lambda^f_{p_s} \right) \right\}. \end{aligned}$$

The Lagrange function can be rewritten as

$$L(Q,q) = \max_{\vec{\lambda}} \left\{ \sum_f U(\lambda^f) - q^f \cdot \lambda^f \right\} \tag{12}$$

$$+ \max_{\vec{\lambda}_{p_s}} \left\{ \sum_{p_s,f} (q^f - Q^f_{p_s}) \cdot \lambda^f_{p_s} \right\} \tag{13}$$

$$+ \max_{\vec{\mu}} \left\{ \sum_{p,p',f} (Q^f_p - Q^f_{p'}) \cdot \mu^f_{p,p'} \right\} \tag{14}$$

$$+ \max_{\vec{r}} \left\{ \sum_{(p,p'),f} (Q^f_p - Q^f_{p'}) \cdot r^f_{(p,p')} \right\}. \tag{15}$$

To obtain (15), we applied the property [15]: $\sum_{p,f} Q^f_p \left( \sum_{p'':(p,p'')\in\mathcal{L}_\mathcal{P}} r^f_{(p,p'')} - \sum_{p':(p',p)\in\mathcal{L}_\mathcal{P}} r^f_{(p',p)} \right) = \sum_{(p,p'),f} (Q^f_p - Q^f_{p'}) r^f_{(p,p')}$.

It can be seen that the resource allocation problem is now decomposed to four subproblems.

- *Flow control*: The Lagrange multiplier $q^f$ can be interpreted as the queue length at the source node of flow $f$. The maximization problem (12) implies the flow control with utility maximization, which determines the input rate at each source node based on the local backlog information $q^f$. Furthermore, the maximization over each flow is decoupled.
- *Traffic distribution at the source nodes*: The maximization problem (13) solves the traffic distribution issue at each source node through local computation. The Lagrange multiplier $Q^f_{p_s}$ can be interpreted as the queue length of each tuple associated with the source node $s(f)$. The new traffic generated enters the queue $q^f$ first and is then distributed to the queues $Q^f_{p_s}$ to enter the tuple-based network. Since $\sum_{p_s:n(p_s)=s(f)} \lambda^f_{p_s} = \lambda_f$, it can be seen that the solution of (13) is to set $\lambda^f_{p^*_s} = \lambda^f$ with $p^*_s = \text{argmax}_{p_s:n(p_s)=s(f)}\{q^f - Q^f_{p_s}\}$; and $\lambda^f_{p_s} = 0$ otherwise. The same solution applies to all flows.
- *In-node traffic transition among tuples*: The maximization problem (14) determines the traffic transitions among tuples within a node. Such internal transitions allow the input traffic to be served through a different tuple (i.e., a channel/ratio combination) from the input tuple for the optimal resource utilization. The Lagrange multiplier $Q^f_p$ can be interpreted as the queue length maintained at tuple $p$ for flow $f$. The solutions of the problem (14) are as follows. For a tuple $p'$ at a node, set $\mu^{f*}_{p*,p'} = \max_{p'':(p',p'')\in\mathcal{L}_\mathcal{P}} w^{c(p',p'')}_{(p',p'')}$ with $(p^*, f^*) = \text{argmax}_{p,f}\{Q^f_p - Q^f_{p'}\}$; and $\mu^f_{p,p'} = 0$ otherwise.
- *Throughput-optimal scheduling:* The solution of (15) gives the feasible flow allocation over each tuple link. When a subgradient search method is used to dynamically solve the problem [15], at each step, the solution of (15) under the interference constraint will be in the form of a back-pressure algorithm for throughput-optimal control, with more details to be discussed below.

It is noteworthy that in the decomposable framework (12)–(15), the subproblems of flow control, traffic distribution, and in-node traffic transition can be locally computed in a distributed manner. The throughput-optimal scheduling problem (15) however requires the global information and centralized computation. Consider a dynamic algorithm with the subgradient search. We add "$(t)$" to related variables to specifically indicate the values determined at the $t$th step. For the maximization problem (15) at the $t$th step, for the tuple link $(p,p')$, we select the flow $f^*$ such that $f^* = \arg\max_f \{Q^f_p(t) - Q^f_{p'}(t)\}$, and set $r^{f^*}_{(p,p')}(t) = w^{c(p,p')}_{(p,p')}$ and $r^f_{(p,p')}(t) = 0$ for $f \neq f^*$. Then, solve (15) as

$$\max_{\vec{r}} \left\{ \sum_{(p,p')\in\mathcal{L}_\mathcal{P}} [Q^{f^*}_p(t) - Q^{f^*}_{p'}(t)]^+ \cdot r^{f^*}_{(p,p')} \right\} \tag{16}$$

$$= \max_{\mathcal{M}_{\mathcal{P}}} \left\{ \sum_{(p,p')\in\mathcal{L}_{\mathcal{P}}} [Q_p^{f^*}(t) - Q_{p'}^{f^*}(t)]^+ \cdot w_{(p,p')}^{c(p,p')} \right\} \quad (17)$$

$$= \max_{\mathcal{M}_{\mathcal{P}}} \left\{ \sum_{(p,p')\in\mathcal{L}_{\mathcal{P}}} Q_{(p,p')}^{f^*}(t) \cdot w_{(p,p')}^{c(p,p')} \right\} \quad (18)$$

where $Q_{(p,p')}^{f^*} = [Q_p^{f^*} - Q_{p'}^{f^*}]^+$ is defined as the backlog of flow $f^*$ over a tuple link $(p,p')$. The expression (18) in fact defines the *tuple-based back pressure* algorithm, with $\mathcal{M}_{\mathcal{P}}$ denoting all the maximal scheduling sets over the tuple-based network.

Let $\lambda^f(t)$, $\lambda_{p_s}^f(t)$, and $\mu_{p,p'}^f(t)$ denote the solutions of the subproblems of flow control, traffic distribution, and in-node traffic transition, respectively, at the $t$th step. With the subgradient method, at the $(t+1)$th iteration, the Lagrange multipliers are updated by

$$Q_p^f(t+1) = \Bigg[ Q_p^f(t) - \sum_{(p,p'')\in\mathcal{L}_{\mathcal{P}}} r_{(p,p'')}^f(t) + \sum_{(p',p)\in\mathcal{L}_{\mathcal{P}}} r_{(p',p)}^f(t) - \sum_{p'':n(p'')=n(p)} \mu_{p,p''}^f(t) + \sum_{p':n(p')=n(p)} \mu_{p',p}^f(t) + \lambda_{p_s}^f(t)\mathbf{1}_{\{p\in p_s\}} \Bigg]^+ \quad (19)$$

$$q^f(t+1) = \left[ q^f(t) - \sum_{p_s:n(p_s)=s(f)} \lambda_{p_s}^f(t) + \lambda^f(t) \right]^+ \quad (20)$$

where $\mathbf{1}_A$ is 1 if event $A$ is true, and 0 otherwise. The convex optimization theory tells that the iterative subgradient method, with the Lagrange multipliers updated by (19) and (20), converges to the optimal solution.

Expressing the tuple-based back-pressure scheduling (18) in a general context independent of the specific flow $f$, we use $\mathcal{M}_{\mathcal{P}}^*(t)$ to denote a tuple-based throughput-optimal schedule in time-slot $t$, which should satisfy

$$\mathcal{M}_{\mathcal{P}}{}^*(t) = \arg\max_{\mathcal{M}_P(t)} \sum_{\ell:\ell\in\mathcal{M}_{\mathcal{P}}(t)} Q_\ell(t) w_\ell^{c(\ell)}. \quad (21)$$

The throughput optimality of the tuple-based back-pressure scheduling (21) can be proved by conducting Lyapunov analysis in the virtual SR-SC network enabled by the tuple-based model. The analysis basically follows the standard procedure given in [15]. Due to the page limit, the proof is omitted here.

*Remark 3:* The tuple-based scheduling (18) theoretically gives the joint optimal solutions of link scheduling, channel assignment, and radio assignment. Note that the scheduling of a tuple link $\ell = ((i,m_i,c),(j,m_j,c))$ can indicate that the link $(i,j)$ is scheduled on channel $c$, where nodes $i$ and $j$ use the radio interfaces $m_i$ and $m_j$, respectively, to serve the link.

*Theorem 1:* In an MR-MC network, the optimal capacity region under the tuple-based throughput-optimal scheduling (21) is equivalent to that under the link-based throughput-optimal scheduling (1) or (2).

*Proof:* Let $\Omega$ and $\Omega_{\mathcal{P}}$ denote the optimal capacity region under the link-based scheduling (1) or (2) and that under the tuple-based scheduling (21), respectively. We then use $\vec{r}$ and $\vec{r}_{\mathcal{P}}$ to denote a feasible link rate vector and a feasible tuple-link rate vector, respectively.

Given a feasible rate vector $\vec{r} \in \Omega$, the throughput-optimal algorithm (2) will generate a maximal schedule $\mathcal{M}(t) = \{\mathcal{M}_c(t)\}$ at each slot. The maximal schedule $\{\mathcal{M}_c(t)\}$ satisfies that: 1) simultaneous transmissions out of the interference ranges of each other are allowed; 2) simultaneous transmissions within the interference range need to occupy dedicated radio interfaces for both the sender and the receiver nodes and use separate channels. That is, within each schedule $\mathcal{M}_c(t)$, each transmission is associated with a certain link $(i,j)$ over a channel $c$, and both the sender node $i$ and the receiver node $j$ occupy a radio interface, say, $m_i$ and $m_j$, respectively. Denote the transmission in the form as $((i,m_i,c),(j,m_j,c))$, which in fact indicates a tuple link. Therefore, the schedule set $\{\mathcal{M}_c(t)\}$ is equivalent to a tuple-based maximal schedule $\mathcal{M}_{\mathcal{P}}(t)$. In the tuple-based network, this schedule $\mathcal{M}_{\mathcal{P}}$ supports a feasible tuple-link rate vector $\vec{r}_{\mathcal{P}} \in \Omega_{\mathcal{P}}$, which is equivalent to the original link rate vector $\vec{r}$.

Given a feasible vector $\vec{r}_{\mathcal{P}} \in \Omega_{\mathcal{P}}$ in the tuple-based virtual SR-SC network, the throughput-optimal algorithm (21) will generate a maximal schedule $\mathcal{M}_{\mathcal{P}}(t)$ at each slot to serve the traffic. A link rate vector $\vec{r}$ can be constructed from $\vec{r}_{\mathcal{P}}$ as follows: Given a link $l' \in \mathcal{L}$, $r_{l'} = \sum_{\ell:l(\ell)=l'} r_\ell$. The link rate vector $\vec{r} = \{r_{l'}\}$ is feasible, that is, $\vec{r} \in \Omega$, as the maximal schedule $\mathcal{M}_{\mathcal{P}}$ indicates all the required resource allocation of link scheduling and channel and radio assignment to support the traffic, referring to Remark 3. ■

### C. Cross-Layer Control With Path Selection

It has been shown that the classic back-pressure algorithm could lead to unnecessarily large delays [16], [20]. In this section, we derive a decomposable cross-layer control to mitigate the end-to-end delay by integrating a network-layer path selection with the link-layer scheduling. Such a joint routing and scheduling algorithm is a specific case within the general cross-layer framework developed above.

Assume that there are $R(f)$ paths from source to destination for flow $f$, and $H_k$ is the number of hops for path $k$, where $k \in \{1,\ldots,R(f)\}$. Furthermore, we define a routing matrix $[\Upsilon_{kp}^f]$; $\Upsilon_{kp}^f = 1$ indicates tuple $p$ is on the path $k$ of flow $f$, and 0 otherwise. Let $A_k^f$ denote the fraction of flow $f$ injected to path $k$. Note that in this section, the input rate $\vec{\lambda}$ is given. We formulate the optimization problem (**P2**)

$$\min_{A_k^f,\vec{r}} \sum_f \sum_{k=1}^{R(f)} \Lambda H_k A_k^f \quad (22)$$

Subject to:

$$\sum_{k=1}^{R(f)} A_k^f \Upsilon_{kp}^f + \sum_{p':(p',p)\in\mathcal{L}_{\mathcal{P}}} r_{(p',p)}^f \le \sum_{p'':(p,p'')\in\mathcal{L}_{\mathcal{P}}} r_{(p,p'')}^f \quad \forall p\in\mathcal{P},\ f\in F \quad (23)$$

$$\sum_{k=1}^{R(f)} A_k^f = \lambda^f, \quad A_k^f \ge 0 \qquad \forall f\in F \quad (24)$$

$$r_{(p,p')}^f \ge 0 \qquad \forall f\in F,\ \forall (p,p')\in\mathcal{L}_{\mathcal{P}} \quad (25)$$

$$\vec{r} \in Co(\mathcal{M}_{\mathcal{P}}) \quad (26)$$

where $\Lambda$ is a positive constant as a control parameter. The objective function (22) is to minimize the traffic incurred within the network to support all commodity flows. We do not need the variable $\vec{\mu}$ in the flow conservation constraint (23) since the routing matrix (at the tuple level) has been given. As suggested

in [16], we use the simplified assumption in constraint (23) that the new arrival packets of flow $f$ are injected to all tuple links along the path of flow $f$ simultaneously, for the convenience of obtaining a path selection algorithm with clean physical meaning.

We introduce $Q_p^f$ as the Lagrange multiplier associated with (23) and obtain the partial Lagrange dual function as

$$L(Q) = \min_{A_k^f} \left\{ \sum_f \sum_{k=1}^{R(f)} A_k^f \left( \Lambda H_k + \sum_p Q_p^f \Upsilon_{kp}^f \right) \right\} \quad (27)$$

$$- \max_{\vec{r}} \left\{ \sum_{f,(p,p')\in\mathcal{L}_\mathcal{P}} \left( Q_p^f - Q_{p'}^f \right) r_{(p,p')}^f \right\}. \quad (28)$$

It is not difficult to see that the solution of $L(Q)$ leads to a joint routing and scheduling algorithm.

- *Step 1: Path selection.* At time-slot $t$, for flow $f$, the input traffic $\lambda^f$ is injected into the path $k^*(f)$ such that

$$k^*(f) = \arg\min_{1\le k\le R(f)} \left\{ \Lambda H_k + \sum_p Q_p^f(t) \Upsilon_{kp}^f \right\} \quad (29)$$

  which is in fact the solution to (27).
- *Step 2: Scheduling*. Apply the tuple-based back-pressure algorithm (18) to schedule the transmissions, which is in fact the solution to (28).

Note that the expression $\Lambda H_k + \sum_p Q_p^f \Upsilon_{kp}^f$ can be interpreted as an indicator of the end-to-end delay, where the hop count $H_k$ is directly related to the end-to-end transmission delay, and $\sum_p Q_p^f \Upsilon_{kp}^f$ indicates the total queuing delay along path $k$. The control parameter $\Lambda$ is to trade off the impact of transmission delay and queueing delay on path selection.

*Remark 4:* A hop-count-based path selection [20] has been exploited in SR-SC networks, but the hop count is not enough to determine the delay in the MR-MC context. For example, consider routing over two connected links with two available channels. There exist four possible paths considering the channel combinations, each of which may give different delay due to the channel-dependent link capacity. For satisfying delay performance, we must exploit the channel dependence in the network-layer path selection, which is enabled by using the tuple-based path in our algorithm. It is worth noting that the work in [16] also considered path-selection-based cross-layer control. However, the link-based scheduling algorithm as summarized in Section II-B could only partially present the input queue information to the network layer, but is incapable of incorporating the per-channel output queues (referring to [16, Sec. IV-B]), thus leading to inaccurate evaluation of the path delay.

When implementing the path-selection-based cross-layer control, we resort to the multidimensional conflict-graph-based capacity planning technique we developed in [28] to generate a set of candidate paths for use in Step 1. The development in Section IV-B demonstrates that the throughput-optimal control is in fact a dynamic implementation of a subgradient search method to solve the MCF problem using convex duality. Thus, if the network is initiated with the optimal dimensioning configuration, as long as the input traffic dynamics are within the capacity region, the online throughput-optimal control will maintain the network stable. Such an offline dimensioning and online control configuration is also consistent with the "first plan, then take care" methodology, recommended by the network management studies for both wireline networks [31] and wireless networks [32]. Seen from another perspective, a basic issue in the cross-layer control framework is how to properly select (from the exponentially many possible paths) the set of candidate paths. Using the paths generated from offline planning provides one approach that can lead to a large network capacity with a relatively small set of paths. The efficiency of this approach will be demonstrated by simulation results in Section VII.

## V. Tuple-Based Greedy Maximal Scheduling

The throughput-optimal control problem (21) under the interference constraint is NP-hard. In this section, we develop a *tuple-based greedy maximal scheduling* as an efficient, low-complexity approximation to the throughput-optimal scheduling and analyze its performance. As the stability and capacity analysis in the context of multiple commodity flows is essentially the same as that in the context of single flow [15], our analysis in the remaining part of this paper will drop the flow index $f$ for convenience.

### A. Tuple-Based GMS Algorithm

The tuple-based model maps an MR-MC network to a virtual SR-SC network, where the GMS developed in the SR-SC context could be directly applied to approximate the throughput-optimal scheduling (21). Specifically, the *tuple-based GMS* starts from the tuple link $\ell$ that has the largest weight $Q_\ell w_\ell^{c(\ell)}$, and proceeds as follows.

- *Step 1*: Pick the tuple link with the largest weight $Q_\ell(t) w_\ell^{c(\ell)}$ and add it into the schedule set $\mathcal{M}_\mathcal{P}(t)$.
- *Step 2*: Remove from the tuple link set $\mathcal{L}_\mathcal{P}$ all the tuple links that interfere with the selected tuple link $\ell$, according the interference model defined in Section III-B.
- *Step 3*: Repeat steps 1 and 2 until the set $\mathcal{L}_\mathcal{P}$ is empty.

It is hard to rigorously compare the capacity region of tuple-based GMS to that of the link-based GMS. A heuristic analysis can show that the tuple-based scheduling will have a similar performance to the link-based scheduling in a homogeneous environment, where the capacity of each link has the same distribution over all channels and each node fairly competes for the transmission opportunities through the same number of radios. In such a homogeneous environment, each link on average will be scheduled with a similar number of parallel transmissions through different channels over separate radios, i.e., tuple links activated simultaneously. Each tuple link associated with the same physical link will have roughly the same average queue length due to the fair resource allocation. Let $X$ denote the average number of scheduled tuple links associated with each physical link (assuming a homogeneous case). Let $Q_l$ represent the total backlog crossing all scheduled tuple links associated with physical link $l$. If we indicate the long-term average case by omitting the notation "$(t)$," we have

$$\mathcal{M}_\mathcal{P}^* = \arg\max_{\mathcal{M}_P} \sum_{\ell:\ell\in\mathcal{M}_\mathcal{P}} Q_\ell w_\ell^{c(\ell)} \approx \arg\max_{\mathcal{M}_P} \sum_{\ell:\ell\in\mathcal{M}_\mathcal{P}} \frac{Q_{l(\ell)}}{X} w_\ell^{c(\ell)}$$

$$= \arg\max_{\{\mathcal{M}_c\}} \frac{1}{X} \sum_{(l,c):l\in\mathcal{M}_c} Q_l w_l^c = \mathcal{M}^*.$$

The last step above is due to the fact that the constant factor $\frac{1}{X}$ will not impact the scheduling decision. The analysis also implies that tuple-based GMS will have similar performance to the link-based GMS. In Section VII, the simulation results do confirm such a kind of behavior.

In the tuple-based network, we have the following lemma on interference degree, which is a fundamental factor determining the capacity region of tuple-based scheduling algorithms.

*Lemma 1:* In the equivalent tuple-based network, the interference degree of a given tuple link $\ell$ satisfies $\mathcal{Y}(\ell) \leq \mathcal{K}(l(\ell)) + 2$, and the network interference degree satisfies $\mathcal{Y} \leq \mathcal{K} + 2$.

*Proof:* The interference degree $\mathcal{Y}(\ell)$ can be obtained by analyzing the maximum number of simultaneous transmissions allowed in $I(\ell)$, when the tuple link $\ell$ turns off its transmission. In an MR-MC network, the co-channel interference conflict relationship (defined in Section III-B) is the same as that in an SR-SC network. Thus, for any subset of *co-channel* tuple links $\{\ell'\} \subset I(\ell)$ being active simultaneously, their corresponding links, forming the set $\{l(\ell')\} \subset I(l(\ell))$, are also free of interference. On the other side, any set of simultaneous transmissions allowed in $I(l(\ell))$ can each spawn a corresponding tuple link in $I(\ell)$ and these spawned tuple links are free of interference. Thus, we can see the maximum cardinality of the subset of interference-free co-channel tuple links as $\max|\{\ell'\}| = \max|\{l(\ell')\}| = \mathcal{K}(l(\ell))$. Furthermore, when the tuple link $\ell$ is turned off, two radios are released at its sender node and receiver node, respectively. In the best situation, these two radio interfaces can allow two more transmissions within $I(\ell)$ using other two separate channels. In total, $\mathcal{Y}(\ell) \leq \mathcal{K}(l(\ell)) + 2$. Therefore, $\mathcal{Y} = \max_\ell\{\mathcal{Y}(\ell)\} \leq \mathcal{K} + 2$. ■

In SR-SC networks, it has been shown that the GMS can achieve a capacity efficiency ratio of $\frac{1}{\mathcal{K}}$ [5]. As the tuple-based network provides an equivalent SR-SC model to the original MR-MC network, we can immediately state that the tuple-based GMS algorithm can achieve a capacity efficiency ratio of $\frac{1}{\mathcal{Y}} \geq \frac{1}{\mathcal{K}+2}$, which is the ratio obtained in [16]. Next, we are to derive a tighter bound of the capacity efficiency ratio resorting to the local pooling factor analysis.

### B. Local-Pooling Factor Analysis

We first summarize the local pooling factor analysis for an SR-SC network developed in [11], using the notations defined in this paper.

*Definition 3 in [11]:* A graph $G(\mathcal{N}, \mathcal{L}, I)$ (with $I$ denoting the interference model) is said to be $\sigma$-*dominant*, if there exist two vectors $\vec{\mu}, \vec{\nu} \in Co(\mathcal{M}_L)$ for a subset of links $L \subset \mathcal{L}$ such that $\sigma\vec{\mu} \succeq \vec{\nu}$, i.e., $\sigma\mu_i \geq \nu_i$ for all $i \in L$. The vectors $\vec{\mu}$ and $\vec{\nu}$ are called $\sigma$-dominant vectors.

The $\sigma$-*dominance* is closely related to the capacity efficiency ratio of the GMS. It turns out that if there exist two $\sigma$-*dominant* vectors $\vec{\mu}, \vec{\nu} \in Co(\mathcal{M}_L)$ such that $\sigma\vec{\mu} \succeq \vec{\nu}$, then we can construct a traffic pattern that: 1) has an arrival rate equal to $\sigma\vec{\mu}$; and 2) induces the service vector of GMS to be $\vec{\nu}$. Thus, the system is unstable at an arrival rate $\sigma\vec{\mu}$ with the GMS control, while the arrival rate $\vec{\mu}$ could have been stabilized under a throughput-optimal policy. Hence, the efficiency ratio of GMS will be no greater than $\sigma$. Some important definitions and propositions developed in [11] are summarized as follows.

*Definition 4 in [11]:* The local-pooling factor $\sigma^*(G)$ of a graph $G(\mathcal{N}, \mathcal{L}, I)$ is the infimum of all $\sigma$ such that the graph $G$ is $\sigma$-dominant, i.e., $\sigma^*(G) := \inf\{\sigma | G \text{ is } \sigma-\text{dominant}\}$.

*Proposition 1 in [11]:* The efficiency ratio $\gamma(G)$ of GMS under a given network graph $G(\mathcal{N}, \mathcal{L}, I)$ is equal to its local-pooling factor $\sigma^*(G)$.

For a subset $L \subset \mathcal{L}$, let $I_L(l) = I(l) \cap L$ denote the set of links in $L$ that interfere with link $l$, and denote the interference degree $d_L(l)$ as the maximum number of links in $I_L(l)$ that can be scheduled at the same time without interfering with each other. Note that notation $d_L(l)$ here explicitly indicates interference degrees discussed in the context for local-pooling factor analysis, while the meaning of $\mathcal{K}(l)$ and $\mathcal{K}$ remains as before. There is the following proposition, which is used as Lemma 2 in this paper.

*Lemma 2 ([11, Proposition 2]):* Given a network graph $G(\mathcal{N}, \mathcal{L}, I)$, assume that a sequence of links $\{l_1, l_2, \ldots, l_{|\mathcal{L}|}\}$ and a sequence of sets $\{L_1, L_2, \ldots, L_{|\mathcal{L}|}, L_{|\mathcal{L}|+1}\}$ with $L_1 = \mathcal{L}$ and $L_{|\mathcal{L}|+1} = \emptyset$ satisfy that $L_{i+1} = L_i \setminus \{l_i\}$ and $d_{L_i}(l_i) \leq d$ for all $1 \leq i \leq |\mathcal{L}|$ with some $d \geq 1$. Then, the local-pooling factor is bounded by $\frac{1}{d}$, i.e., $\sigma^* \geq \frac{1}{d}$.

We can immediately see that the network degree $\mathcal{K}$ satisfies the conditions set in Lemma 2 as a proper $d$ value, which then gives the capacity efficiency ratio of $\frac{1}{\mathcal{K}}$ that has been available in the literature. It is shown in [11] that Lemma 2 can give a tighter bound on the GMS capacity efficiency ratio due to the *edge effect*. When the links are selected according to Lemma 2 each step, link $l_i$ might be an edge link in the set $L_i$; the edge link normally has an interference degree less than $\mathcal{K}$, so Lemma 2 can generate a $d < \mathcal{K}$ and thus a larger capacity efficiency ratio than $1/\mathcal{K}$.

Given an SR-SC network $G(\mathcal{N}, \mathcal{L}, I)$, when MR-MC networking is applied over $G$, we consider the interference model $I$ is extended to $I_\mathcal{P}$ according to Section III-B. The equivalent tuple-based network graph is then denoted as $G_\mathcal{P}(\mathcal{P}, \mathcal{L}_\mathcal{P}, I_\mathcal{P})$. As $G_\mathcal{P}(\mathcal{P}, \mathcal{L}_\mathcal{P}, I_\mathcal{P})$ provides an SR-SC context and the interference model $I_\mathcal{P}$ defines the interference degrees and maximal schedules, all the definitions and propositions summarized above regarding the local pooling factor can be applied to $G_\mathcal{P}(\mathcal{P}, \mathcal{L}_\mathcal{P}, I_\mathcal{P})$. We can immediately have Lemma 3 and then prove Theorem 2 in the following.

*Lemma 3:* The capacity efficiency ratio $\gamma(G_\mathcal{P})$ of a tuple-based GMS in a tuple-based network graph $G_\mathcal{P}(\mathcal{P}, \mathcal{L}_\mathcal{P}, I_\mathcal{P})$ is equal to its local-pooling factor $\sigma^*_\mathcal{P}(G_\mathcal{P})$.

*Theorem 2:* Consider the network $G(\mathcal{N}, \mathcal{L}, I)$ and the equivalent tuple-based network $G_\mathcal{P}(\mathcal{P}, \mathcal{L}_\mathcal{P}, I_\mathcal{P})$. If by the link selection sequence defined in Lemma 2, the local-pooling factor of $G(\mathcal{N}, \mathcal{L}, I)$ is bounded as $\sigma^* \geq \frac{1}{d}$, the local pooling factor of $G_\mathcal{P}(\mathcal{P}, \mathcal{L}_\mathcal{P}, I_\mathcal{P})$ is then bounded as $\sigma^*_\mathcal{P} \geq \frac{1}{d+2}$. That is, the tuple-based GMS can achieve a capacity efficiency ratio at least $\frac{1}{d+2}$.

*Proof:* For the network $G(\mathcal{N}, \mathcal{L}, I)$, assume the link selection sequence by Lemma 2 is $S = \{l_1, l_2, \ldots, l_{|\mathcal{L}|}\}$. All the tuple links spawned by the link $l_i \in S$ in the sequence form a set $B_i \subset \mathcal{L}_\mathcal{P}$. In the equivalent network $G_\mathcal{P}(\mathcal{P}, \mathcal{L}_\mathcal{P}, I_\mathcal{P})$, we construct a tuple-link selection sequence as follows. We first pick tuple links from the set $B_1$; within set $B_1$, the tuple links can be picked by any order. When all the tuple links in $B_1$ are selected, then the tuple links in set $B_2$ will be selected. The procedure is then repeated over next set until all tuple links are selected. Record such sequence of tuple-link selection as $\{\ell_1, \ell_2, \ldots, \ell_{|\mathcal{L}_\mathcal{P}|}\}$. Further define the sequence of sets $\{E_1, E_2, \ldots, E_{|\mathcal{L}_\mathcal{P}|}, E_{|\mathcal{L}_\mathcal{P}|+1}\}$ with $E_1 = \mathcal{L}_\mathcal{P}$ and $E_{i+1} = E_i \setminus \{\ell_i\}$ for $1 \leq i \leq |\mathcal{L}_\mathcal{P}|$. If for all the link $l_i \in S$, $d_{L_i}(l_i) \leq d$ with $1 \leq i \leq |\mathcal{L}|$, we then have $d_{E_i}(\ell_i) \leq d + 2$ according to Lemma 1. Applying Lemma 2 to the equivalent

SR-SC network $G_{\mathcal{P}}(\mathcal{P}, \mathcal{L}_{\mathcal{P}}, I_{\mathcal{P}})$, we can obtain $\sigma^*_{\mathcal{P}} \geq \frac{1}{d+2}$, which is also a lower bound of the capacity efficiency ratio of the tuple-based GSM by Lemma 3. ■

A very popular model for studying wireless network is the geometric unit-disc graph with the 2-hop interference model [11]. When such a model is extended to the MR-MC context, we have the following corollary based on Theorem 2.

*Corollary 1:* Given an MR-MC network, if the underlying SR-SC infrastructure (ignoring the configuration of multiple radios and multiple channels) is a geometric unit-disc graph model with the 2-hop interference model, the worst-case efficiency ratio of the tuple-based GMS is $\frac{1}{8}$.

*Proof:* In [11], a technique is developed to sequentially select the *leftmost* link, denoted as $l^*$, from the unit-disc graph. It is further proved that each leftmost link has an interference degree $d(l) \leq 6$ and then achieve a recurrent interference degree $d \leq 6$. Further by using Theorem 2, the tuple-based GMS can achieve a capacity efficiency ratio of $\frac{1}{d+2} \geq \frac{1}{8}$. ■

Note that it has been shown that the network interference degree of a geometric unit-disc graph model with the 2-hop interference model is $\mathcal{K} = 8$ [12]. The capacity efficiency ratio by the analysis in [16] is $\frac{1}{\mathcal{K}+2} = \frac{1}{10}$, worse than the local-pooling factor-based ratio of $\frac{1}{8}$.

## VI. Tuple-Based Maximal Scheduling

The tuple-based model also allows us to readily extend the distributed maximal scheduling in SR-SC networks [10] to the MR-MC networks. We define a tuple link $\ell$ as backlogged if $Q_\ell > \Gamma w_\ell^{c(\ell)}$, where $\Gamma \geq 1$. For any tuple link $\ell$ that is backlogged, either of the following is true.

- The tuple link $\ell$ is activated on channel $c(\ell)$.
- Another tuple link $\ell'$, which is backlogged and conflicts with $\ell$, is scheduled on channel $c(\ell')$.

According to our Remark 3, the maximal schedule set generated by the tuple-based MS can jointly indicate link scheduling and channel and radio assignments.

Note that a backlogged link was defined with $\Gamma = 1$ in [10]. Our generalized definition, allowing $\Gamma$ to be an integer $\geq 1$, will bring us the flexibility to control the impact of communication overhead in a distributed implementation of the maximal scheduling. We define one packet as what a tuple link can transmit at its full capacity in one unit time. Each scheduled backlogged tuple link can then guarantee $\Gamma$ packets for transmission within each scheduling period.

By extending the capacity analysis in the SR-SC context [10], we can have the following theorem.

*Theorem 3:* A tuple link $\ell$ is defined as backlogged and eligible for scheduling if $Q_\ell > \Gamma w_\ell^{c(\ell)}$, with $\Gamma \geq 1$. The tuple-based MS can achieve a capacity efficiency ratio of $\frac{1}{Y} \geq \frac{1}{\mathcal{K}+2}$.

*Proof:* Theorem 1 has proved that the optimal capacity region under the tuple-based throughput-optimal scheduling is equivalent to that under the link-based throughput-optimal scheduling. Thus, we here study the capacity region with the tuple-based model.

According to the capacity region defined in Section II, it suffices to consider the scenario that there is a single-hop flow over each tuple link. Note that a certain tuple $p$ only needs to maintain a queue for each outgoing flow of which it is the source; for each incoming flow ending at $p$, it will consume the traffic and present a queue of length zero. For the convenience of analysis, we adopt a tuple-link-based model. Let $A_\ell(t)$ and $D_\ell(t)$ denote the number of arrivals to and departures from tuple link $\ell = (p, p')$ in slot $t$, associated with flow $f_\ell$. The dynamics of the link queue $Q_\ell(t)$ are given by

$$Q_\ell(t+1) = Q_\ell(t) + A_\ell(t) - D_\ell(t). \tag{30}$$

In the tuple-link-based model, $Q_\ell(t)$ is in fact equivalent to the queue $Q_p^{f_\ell}$ maintained at the sender tuple $p$, and the departure traffic $D_\ell(t)$ is consumed at the receiver tuple $p'$.

We assume that the arrival processes are stationary with $r_\ell = E(A_\ell(t))$ and have bounded covariance [10]. Let $\pi_\ell(t)$ be an indicator function, which takes the value of 1 if tuple link $\ell$ is scheduled in slot $t$, and the value of 0 otherwise. We then have $D_\ell(t) = \pi_\ell(t) w_\ell^{c(\ell)}$. Also, by the maximal scheduling policy, we have

$$\sum_{k:\ell_k \in I(\ell)} \pi_{\ell_k} \geq 1 \qquad \forall \ell. \tag{31}$$

We are to show that when the link flow rate vector $\vec{r}_{\mathcal{P}}$ satisfies

$$\sum_{k:\ell_k \in I(\ell)} \frac{r_{\ell_k}}{w_{\ell_k}^{c(\ell_k)}} < 1 \qquad \forall \ell \tag{32}$$

the tuple-based MS can maintain the network stable. According to [10], we can define the Lyapunov function

$$V(t) = \sum_\ell \frac{Q_\ell(t)}{w_\ell^{c(\ell)}} \left( \sum_{k:\ell_k \in I(\ell)} \frac{Q_{\ell_k}(t)}{w_{\ell_k}^{c(\ell_k)}} \right). \tag{33}$$

Following the steps that were taken in the proof of [10, Theorem 1], we can get

$$\begin{aligned}
&V(t+1) - V(t) \\
&= 2 \sum_\ell \frac{Q_\ell(t)}{w_\ell^{c(\ell)}} \left( \sum_{k:\ell_k \in I(\ell)} \frac{Q_{\ell_k}(t+1) - Q_{\ell_k}(t)}{w_{\ell_k}^{c(\ell_k)}} \right) \\
&\quad + \sum_\ell \frac{Q_\ell(t+1) - Q_\ell(t)}{w_\ell^{c(\ell)}} \left( \sum_{k:\ell_k \in I(\ell)} \frac{Q_{\ell_k}(t+1) - Q_{\ell_k}(t)}{w_{\ell_k}^{c(\ell_k)}} \right) \\
&= 2 \sum_\ell \frac{Q_\ell(t)}{w_\ell^{c(\ell)}} \left( \sum_{k:\ell_k \in I(\ell)} \left( \frac{A_{\ell_k}(t)}{w_{\ell_k}^{c(\ell_k)}} - \pi_{\ell_k}(t) \right) \right) \\
&\quad + \sum_\ell \left( \frac{A_\ell(t)}{w_\ell^{c(\ell)}} - \pi_\ell(t) \right) \left( \sum_{k:\ell_k \in I(\ell)} \left( \frac{A_{\ell_k}(t)}{w_{\ell_k}^{c(\ell_k)}} - \pi_{\ell_k}(t) \right) \right).
\end{aligned}$$

Using the bounded second moment assumption and the fact that the number of departures from each interference set is bounded, we get

$$\begin{aligned}
&E\left(V(t+1) - V(t) | Q_\ell(t)\right) \\
&\leq 2 \sum_\ell \frac{Q_\ell(t)}{w_\ell^{c(\ell)}} \left( \sum_{k:\ell_k \in I(\ell)} \frac{r_{\ell_k}(t)}{w_{\ell_k}^{c(\ell_k)}} - \sum_{k:\ell_k \in I(\ell)} \pi_{\ell_k}(t) \right) + B \\
&= \sum_{\ell: Q_\ell(t) > 0} \frac{Q_\ell(t)}{w_\ell^{c(\ell)}} \left( \sum_{k:\ell_k \in I(\ell)} \frac{r_{\ell_k}(t)}{w_{\ell_k}^{c(\ell_k)}} - \sum_{k:\ell_k \in I(\ell)} \pi_{\ell_k}(t) \right) + B.
\end{aligned}$$

Considering that only backlogged tuple links are considered for scheduling, i.e., $\pi_\ell(t) = 0$ for all unbacklogged tuple links, and using (31), we further have

$$\begin{aligned}
&E\left(V(t+1) - V(t) | Q_\ell(t)\right) \\
&\leq 2 \sum_{\ell: Q_\ell(t) > \Gamma w_\ell^{c(\ell)}} \frac{Q_\ell(t)}{w_\ell^{c(\ell)}} \left( \sum_{k:\ell_k \in I(\ell)} \frac{r_{\ell_k}(t)}{w_{\ell_k}^{c(\ell_k)}} - 1 \right)
\end{aligned}$$

$$
\begin{aligned}
&+2\sum_{\ell:Q_\ell(t)\le\Gamma w_\ell^{c(\ell)}}\Gamma\sum_{k:\ell_k\in I(\ell)}\frac{r_{\ell_k}(t)}{w_{\ell_k}^{c(\ell_k)}}+B\\
&\le 2\sum_{\ell:Q_\ell(t)>\Gamma w_\ell^{c(\ell)}}\frac{Q_\ell(t)}{w_\ell^{c(\ell)}}\left(\sum_{k:\ell_k\in I(\ell)}\frac{r_{\ell_k}(t)}{w_{\ell_k}^{c(\ell_k)}}-1\right)+B_1\\
&\le -2\epsilon\sum_{\ell:Q_\ell(t)>\Gamma w_\ell^{c(\ell)}}\frac{Q_\ell(t)}{w_\ell^{c(\ell)}}+B_1
\end{aligned}
$$

where $B$, $B_1 > 0$ are some constants and

$$\epsilon = 1-\max_{\ell}\sum_{k:\ell_k\in I(\ell)}\frac{r_{\ell_k}(t)}{w_{\ell_k}^{c(\ell_k)}}.$$

We have $\epsilon > 0$, under the constraint (32).

The above result shows that the drift of the Lyapunov function will be negative when the queue lengths are large enough. Thus, the whole system is stable, i.e., $\limsup_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}E(Q_\ell(t)) < \infty$, $\forall \ell \in \mathcal{L}_{\mathcal{P}}$. It is also known that a necessary condition for network stability under any scheduling policy is

$$\sum_{k:\ell_k\in I(\ell)}\frac{r_{\ell_k}}{w_{\ell_k}^{c(\ell_k)}} < \mathcal{Y} \le \mathcal{K}+2 \qquad \forall \ell. \tag{34}$$

Based on the sufficient condition (32) and the necessary condition (34), we can obtain the capacity efficiency ratio of $\frac{1}{\mathcal{Y}} \ge \frac{1}{\mathcal{K}+2}$. ■

While this paper focuses on the link rate vector-based capacity region, the analysis conducted in the proof of Theorem 3 can also be extended to analyze the network layer capacity region with multihop routes. According to [10], the way to achieve queue stability in a multihop network system is to introduce *regulators* in the system. More discussions are given in the Appendix.

### A. Communication Overhead

Implementing the maximal scheduling in a fully distributed manner relies on message exchanges among the nodes, which will incur considerable communication overhead, especially with the interference from other simultaneous transmissions. In [19] and [23], a randomized distributed algorithm is developed for the distributed implementation of MS accounting for message exchanges, under the 1-hop and 2-hop interference models. We here show that the randomized algorithm and the associated communication overhead analysis in [19] can be extended to MR-MC networks.

*1) Randomized Distributed Algorithm:* With the algorithm developed in [19], a time-slot consists of two periods for scheduling and data transmission, respectively. The scheduling period is used to choose a set of noninterfering links, and the data transmission period is used to transmit data packets over the chosen links. The scheduling period is further divided into mini-slots. The transmission of control messages takes place in rounds, each occupying a mini-slot.

Let $N(u)$ denote the set of nodes directly connected with node $u$, i.e., $N(u) = \{v \in \mathcal{N} | (u,v) \in \mathcal{L}\}$, and define $\delta = \max_{u\in\mathcal{N}}|N(u)|$. At each time-slot, each node first updates the backlog for each link it maintains based on the scheduling in the previous time-slot. Then, the algorithm uses two hierarchical loops to compute the schedule in a distributed fashion: $\lceil \mathbb{C}_P \log|\mathcal{N}| \rceil$ phases, and $\lceil \mathbb{C}_I \delta \log|\mathcal{N}| \rceil$ iterations for each phase, where $\mathbb{C}_P$ and $\mathbb{C}_I$ are constants whose values are determined properly to ensure reliable exchange of control messages. At each iteration, each node $u$ tries to transmit a request-to-send (RTS) message with a properly designed probability if the node is eligible, i.e., blocked and not scheduled yet in this slot. If the RTS message is responded with a clear-to-send (CTS) message by the receiver $v$, then the link $(u,v)$ is included into the schedule. When all the iterations finish, if a node is involved in a newly scheduled link (as a sender or a receiver), it then updates the new scheduling information to its neighbors through a reliable broadcast scheme. The reliable broadcasting is achieved in a randomized manner: The node broadcasts the message for $\lceil \mathbb{C}_B \delta \log|\mathcal{N}| \rceil$ iterations, with a properly designed probability at each iteration. When all phases end, if node $u$ has a link $(u,v)$ scheduled, it transmits data packets to $v$ during the data transmission period.

It is proved in [19] that if the constants $\mathbb{C}_P$, $\mathbb{C}_I$, and $\mathbb{C}_B$ are appropriately chosen, the randomized algorithm returns a maximal scheduling with high probability. It is not difficult to see that the randomized distributed algorithm in total needs $\Theta(\delta \log^2|\mathcal{N}|)$ rounds of local message exchanges (to finish the two hierarchical loops), as the communication overhead.

The length of the control message was not explicitly considered in [19]. In fact, small-size packets are sufficient to deliver the control messages for MS implementation. Specifically, the RTS/CTS packets just need to carry a couple of flag bits for handshaking purpose, in addition to standard packet headers. At each slot, a node associated with a link scheduled in the slot just needs to update its queue length of the commodity flow being scheduled (i.e., $Q_u^f$) through broadcasting to its neighbors, so that the neighbors can update corresponding link backlogs for use in the next time-slot. Under the 2-hop interference model, a node needs to relay the scheduling information it received to the next hop [19], [23]; it will be sufficient that the relayed information just contains the address information to indicate the nodes being scheduled.

*2) Randomized Algorithm in MR-MC Networks:* The randomized distributed algorithm and associated communication overhead analysis are applicable in the MR-MC context too, implementing either the link-based MS in [16] or our tuple-based MS. The essential point of the randomized algorithm is that the constants $\mathbb{C}_P$, $\mathbb{C}_I$, and $\mathbb{C}_B$ are appropriately chosen, so that the RTS/CTS message exchanges and scheduling information updating in the neighborhood can be delivered successfully with high probability, through enough rounds of random transmissions. Thus, the schedule formed at the end of the scheduling period is maximal with high probability. While a successful RTS/CTS message exchange schedules one link in the SR-SC context, the same pair of nodes can schedule all eligible link–channel pairs. For example, a node pair $(u,v)$ (after a successful exchange of RTS/CTS messages) can schedule $\min(M_u, M_v, C)$ parallel transmissions. By this manner, we can see that all the nodes, which generate a maximal scheduling in the SR-SC context by the randomized distributed algorithm, will also generate a maximal scheduling in the MR-MC context since all eligible parallel transmissions over separate radios explored will satisfy the multichannel maximal scheduling or tuple maximal scheduling criteria. The communication overhead is therefore in the order of $\Theta(\delta \log^2|\mathcal{N}|)$ too.

When the tuple-based MS is implemented by the randomized algorithm, the control message packet size can still remain small, compared to the SR-SC case. The RTS/CTS messages can remain the same size for handshaking between nodes. If a pair of nodes $(u, v)$ are matched to schedule all the eligible tuple links, the scheduling update message broadcasted from node $u$ or $v$ needs to contain $\min(M_u, M_v, C)$ queue lengths that are associated with the scheduled tuple links. In the case of a 2-hop interference model, the relayed scheduling update message just needs to add some bits to indicate the radios and channels associated with those scheduled tuple nodes, in addition to the node address information that is also included in the SR-SC case. It can be seen that a packet with a payload of tens of bytes should be enough to carry information for all kinds of control messages mentioned above.

When the link-based MS, as summarized in Section II-B, is implemented by the randomized algorithm, the control message packet will need a larger size compared to the tuple-based MS. The stage 1 of the algorithm requires the link capacity and queue length information for all the links within the interference region $I(l)$ over all possible channels. To support such an operation under a 2-hop interference model, each node needs to not only broadcast the transmission capacity and queue length for all the link–channel pairs it maintains, but also relay such information it collects from its neighbors. A control message packet should be long enough to relay such information, which contains $\Theta(\bar{E}C\bar{I})$ queue lengths and link capacities. Note that $\bar{E}$ and $\bar{I}$ denote the average node degree and the average size of an interference set, respectively.

With each slot containing $\Theta(\delta \log^2 |\mathcal{N}|)$ mini-slots for message exchange, the throughput will depend on the length of the transmission period within each slot. Our tuple-based MS can adjust the transmission period through controlling how a backlogged tuple link is defined. According to Theorem 3, if each tuple link is defined as backlogged when $Q_\ell > \Gamma w_\ell^{c(\ell)}$ with $\Gamma$ as an integer $\geq 1$, at each slot, the scheduled tuple link can transmit $\Gamma$ packets. With the length of the scheduling period determined, a larger $\Gamma$ leads to a longer transmission period and thus mitigates the impact of the scheduling period on the data throughput. Theorem 3 guarantees the scalability when we pick a large $\Gamma$.

### B. Computation Overhead

Referring to Section II-B, the link-based MS consists of two stages. If we use $\bar{E}$ to denote the average node degree, i.e., the average number of links incident to a node, it can be seen from (3) that the major computation overhead incurred in stage 1 of the algorithm is in $\Theta(\bar{E}C\mathcal{K} + 2\bar{E}^2C^2)$ division operations. The main computation overhead in stage 2 is $\Theta(\bar{E}C)$ comparison operations to examine whether the link–channel pairs associated with the node are backlogged or not.

Our tuple-based MS algorithm does not need the operations as incurred in stage 1 of the link-based MS. Let $\bar{M}$ denote the average number of radios installed at each node. Under the tuple-based MS, each node just needs $\Theta(\bar{E}\bar{M}^2C)$ comparison operations to examine whether the tuple links associated with the node are backlogged or not.

### C. Implementation of the Cross-Layer Control

The multiprotocol label switching (MPLS) technique [33] can be applied to facilitate the implementation of the cross-layer control with path selection. With MPLS, the paths generated from the offline planning can be deployed as virtual circuits. Path selection at the source node will be implemented by assigning all packets belonging to a commodity flow the same MPLS label indicating the selected path.

When the tuple-based MS is used with the path selection, for collecting the queue lengths to calculate the end-to-end delay metric (29), a scheduling update message broadcast from a node can now include the accumulated queue lengths over all the downstream hops for every tuple-based path (deployed by MPLS) passing it. When an upstream node receives such a message, it will then update its accumulated queue lengths for related paths by adding its local queue length to the received accumulated downstream queue length. This upstream node will then further broadcast its updated accumulated queue length in its scheduling update message. In such a manner, the source node just needs to do the path selection based on the accumulated queue lengths maintained locally.

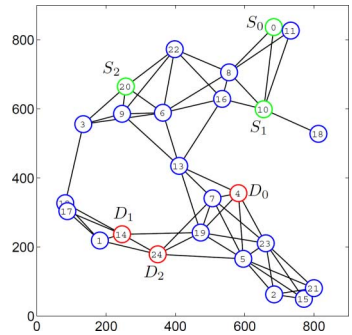


Fig. 2. Random topology.

## VII. Simulation Results

We developed C codes to implement all the scheduling algorithms considered. We construct a random topology with 25 nodes deployed in a $900 \times 900$ m$^2$ area, as shown in Fig. 2. The transmission range and interference range of each node is set to 250 and 500 m, respectively. The link capacity over each channel is uniformly selected in the range $[0.1, 1]$ in each slot to simulate the channel diversity. Each node is equipped with an infinite-size buffer. In the simulation of distributed MS, we omit the communication overhead so that we can clearly observe the performance difference due to the link-based scheduling and the tuple-based scheduling.

### A. Comparison of Scheduling Algorithms

In this experiment, we compare the tuple-based greedy maximal scheduling (TGMS), the tuple-based distributed maximal scheduling (TDMS), and the link-based greedy maximal scheduling (LGMS) [16]. We randomly pick 20 directed links within the topology to deploy single-hop flows. The input rate of each flow is the same, denoted as $\lambda$ with the unit of packets/slot. We set $\Gamma = 1$ for implementing the TDMS.

In Fig. 3, we plot the average backlog at all source nodes versus the input rate $\lambda$. When the input rate increases to a certain value, the average queue length grows sharply. The turning point of a curve indicates the capacity region of the corresponding scheduling algorithm. It is obvious that both TGMS and LGMS achieve a larger capacity region than the TDMS, as expected. An interesting observation is that TGMS and LGMS have very similar performance. The reason is that our setting

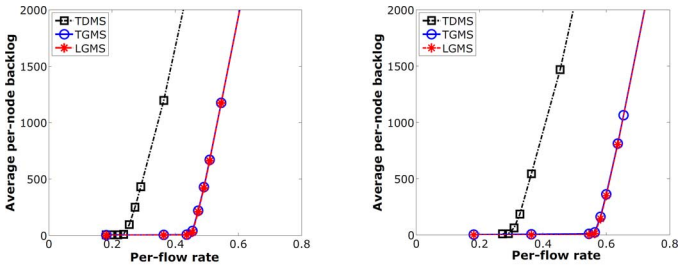



Fig. 3. Average backlog with single-hop traffic under different scheduling algorithms. (a) Two radios and four channels. (b) Three radios and five channels.

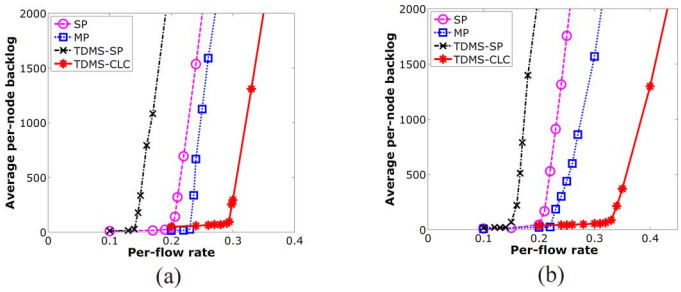



Fig. 4. Average backlog with multihop traffic under the cross-layer control algorithms. (a) Two radios and four channels. (b) Three radios and five channels.

here provides a kind of homogeneous environment. Our analysis in Section V-A does expect such similar performance in a homogeneous context, which is now confirmed by simulations. Fig. 3(a) and (b) presents scheduling performance under two different resource configurations, and a larger capacity region is observed when more radios and channels are available. More detailed analysis can be found in [28] on how the number of radios and channels impact the network capacity.

### *B. Performance With Cross-Layer Control*

Three multihop flows are considered as shown in Fig. 2, where the source and destination nodes for flow $i (i = 1, 2, 3)$ are denoted as $S_i$ and $D_i$ respectively. We gradually increase the flow input rates and observe the average per-node backlog (averaged over those nodes involved in flow transmissions) to examine the capacity region. We compare four algorithms, the single-path (SP) and multipath (MP) algorithms developed in [16] and our TDMS algorithm in the single-path setting (TDMS-SP) and in the cross-layer control setting (TDMS-CLC). The cross-layer control algorithm with path selection is presented in Section IV-C. For TDMS-CLC, the candidate paths at the tuple link level are generated with our capacity planning technique developed in [28]. For a fair comparison, the set of link-based paths are extracted from this set of tuple-based paths to be used in the MP algorithm. The shortest path between each commodity source-destination pair is used for the SP algorithm, and all the tuples associated with the shortest path are used for TDMS-SP algorithm for a fair comparison.

As illustrated in Fig. 4, the MP algorithm has a larger capacity region than SP, showing the benefit of cross-layer control. The TDMS-SP is worse than SP; it is because TDMS-SP schedules tuples purely based on the maximal principle without checking the channel-dependent capacity, but SP does consider the channel quality. Please note that our TDMS-CLC has an obviously larger capacity region than the MP based on

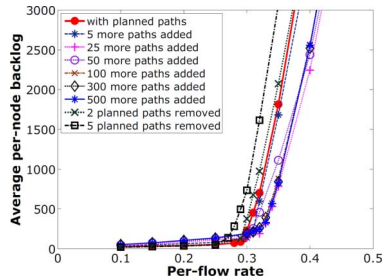



Fig. 5. Average backlog with different number of paths involved in cross-layer control.

the link-based MS. The better performance of TDMS-CLC confirms our analysis in Remarks 2 and 4 that the tuple-based model can indicate backlog information to the upper layer more accurately. Note that a larger capacity region also indicates a better delay performance on average, credited to the good delay performance within the region.

We further evaluate the efficiency of using offline planning paths in cross-layer control. For the example shown in Fig. 2, the offline optimization generates four, six, and five tuple-based paths for commodity flows 1, 2, and 3 respectively. We simulate a series of scenarios by gradually adding more paths (out of the planned path set) into the cross-layer control framework and measure the average per-node backlog (to examine the capacity region) in each scenario. For the case of two radios at each node and four available channels, the capacity region curves associated with different number of additional paths are presented in Fig. 5, where all the additional paths are evenly applied to the three commodities. We can see that including more paths in the cross-layer control does not significantly enlarge the capacity region compared to that under the planned paths; with 500 more paths added, the capacity region (defined by the turning point) increases just around 10%. We also investigate the scenarios when two or five planned paths are removed, where the capacity region obviously reduces; such a behavior indicates that the capacity region is more sensitive on the optimally planned paths. In summary, the simulation results validate that offline planning results ensure a large capacity region in cross-layer control with a small set of paths.

## VIII. Related Work

The common idea for addressing the coupled resource allocation issues in MR-MC wireless networks is to define fine-granularity queues or virtual links, so that the channel/radio assignment issue can be transformed into scheduling of those fine-granularity entities. The studies in [16] and [17] maintain per-channel queues, also termed as link–channel pairs, for each link. As analyzed in this paper, the link–channel-pair-based model has limitations in enabling a decomposable or efficient cross-layer control framework. The work in [24] proposes to transform an MR-MC link into multiple single-radio multichannel (SR-MC) links for scheduling performance analysis. The SR-MC link model is hard to be used in a decomposable cross-layer framework either. The studies in [25] and [26] define a resource configuration mode as a virtual link. While the virtual link technique has been applied in studying the throughput-optimal scheduling for single-hop flows, it can hardly be used in a general utility optimization framework over a multihop MR-MC network: A virtual link, indicating a

resource configuration rather than a physical queue, cannot be used to express the basic flow conservation constraint at a node.

To the best of our knowledge, the tuple-based model is the first fine-granularity model that can explicitly indicate resource allocation in all dimensions of link, radio, and channel. We initially developed the tuple-based model in [28], where the model was constructed with focus on links for the development of a multidimensional conflict graph. In this paper, the tuple-based model is fully generalized into a virtual SR-SC network consisting of NRC tuple nodes and tuple links, where the development of a fully decomposable cross-layer control framework for MR-MC networks is possible.

Optimal capacity planning is another fundamental resource allocation issue complementary to the online scheduling issue. In the SR-SC context, the main-thread approach for wireless network capacity planning is to formulate an LP MCF problem, augmented with constraints derived from a link conflict graph [9], [27]. The conflict graph tool did not achieve the similar popularity in MR-MC networks because the existing tools [29], [30] are not sufficient for fully describing the conflict relations in MR-MC networks, competing for both radios and channels. In [28], we developed the multidimensional conflict graph (MDCG), which enables an LP MCF formulation for an MR-MC network to jointly solve the scheduling, channel/radio assignment, and routing for optimal capacity.

While we focus on a time-slotted system in this paper, random access-based throughput-optimal algorithms are a recent hot topic, referring to [18], [21], and the references therein. The tuple-based model paves the way of exploiting the advances in random-access-based algorithms by considering each tuple node as the entity to contend for the channel.

## IX. Conclusion

In this paper, we conduct a systematic study of GMS and MS algorithms in MR-MC wireless networks by developing a novel virtual SR-SC network model equivalent to the original MR-MC network. Such a model facilitates the derivation of a tuple-based back-pressure algorithm for throughput-optimal control in MR-MC wireless networks and enables the tuple-based GMS and MS scheduling as low-complexity approximation algorithms with guaranteed performance. Compared to the existing link-based algorithms, the tuple-based modeling has significant advantages in enabling a fully decomposable cross-layer control framework. We for the first time extend the local-pooling factor analysis to study the capacity efficiency ratio of the GMS in MR-MC networks and obtain a lower bound that is much tighter than those known in the literature.

## Appendix
## Network Layer Capacity Region of the Maximal Scheduling

A regulator is introduced for each flow using a link such that the burstiness of the traffic is regulated before entry into the node, ensuring the queue stability under a sufficient condition. With the clear context here, we redefine the notation $H_\ell^f$ as an indicator function to indicate whether the tuple link $\ell$ is on the path of commodity flow $f$. The flow over each separate path can be considered as a separate commodity flow. By combining the regulators proposed in [10] and the tuple-based maximal scheduling into a regulated maximal scheduling, we can have the following theorem.

*Theorem 4:* A tuple link $\ell$ is defined as backlogged and eligible for scheduling if it holds at least one flow queue with $Q_\ell^f > \Gamma w_\ell^{c(\ell)}$ and $\Gamma \geq 1$. For a multihop network, if the multicommodity flow vector $(\lambda^1, \ldots, \lambda^f, \ldots, \lambda^F)$ satisfies

$$\sum_{k:\ell_k \in I(\ell)} \frac{\sum_f \lambda^f H_{\ell_k}^f}{w_{\ell_k}^{c(\ell_k)}} < 1 \qquad \forall \ell \tag{35}$$

the network is queue length stable under regulated maximal scheduling.

Theorem 4 can be approved by incorporating the technique of handling a general $\Gamma \geq 1$, as demonstrated in the proof of Theorem 3, into the proof technique given in [10, Sec. V-B]. Due to the page limit, the full proof is omitted here. We however would like to highlight how the queues maintained at each tuple are mapped to the tuple-link-based model for the proof. Each nondestination tuple maintains an output queue $Q_\ell^f$ to buffer the packets for every flow $f$ to be delivered over the outgoing tuple link $\ell$; if not being a source tuple, it will also maintain an input queue $Y_\ell^f$ (corresponding to each $Q_\ell^f$), which buffers the incoming flow-$f$ packets from the tuple link immediately preceding $\ell$ and serves as the regulator to shape the traffic entering the output queue $Q_\ell^f$. The maximal scheduling is applied to the queues $Q_\ell^f$ over all the tuple links. Another detail is that when the regulator applies the probability-based traffic shaping [10], every $\Gamma$ packets need to be controlled as a nonseparable package.

## References


[1] M. Alicherry, R. Bhatia, and L. Li, "Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks," in *Proc. ACM MobiCom*, Aug. 2005, pp. 58–72.

[2] G. Wunder and C. Zhou, "Queueing analysis for the OFDMA downlink: throughput regions, delay and exponential backlog bounds," *IEEE Trans. Wireless Commun.*, vol. 8, no. 2, pp. 871–881, Feb. 2009.

[3] Y. Shi and Y. T. Hou, "A distributed optimization algorithm for multi-hop cognitive radio networks," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 1512–1520.

[4] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Autom. Control*, vol. 4, no. 12, pp. 1936–1948, Dec. 1992.

[5] X. Lin and N. B. Shroff, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," *IEEE/ACM Trans. Netw.*, vol. 14, no. 2, pp. 302–315, Apr. 2006.

[6] P. Chaporkar, K. Kar, and S. Sarkar, "Achieving queue length stability through maximal scheduling in wireless networks," presented at the Inf. Theory Appl. Inaugural Workshop, Feb. 2006.

[7] A. Kabbani, T. Salonidis, and E. W. Knightly, "Distributed low-complexity maximum-throughput scheduling for wireless backhaul networks," in *Proc. IEEE INFOCOM*, May 2007, pp. 2063–2071.

[8] A. Dimakis and J. Walrand, "Sufficient conditions for stability of longest-queue-first scheduling: second-order properties using fluid limits," *Adv. Appl. Probab.*, vol. 38, no. 2, pp. 505–521, 2006.

[9] V. S. Anil Kumar, M. V. Marathe, and S. Parthasarathy, "Algorithmic aspects of capacity in wireless neworks," in *Proc. ACM SIGMETRICS*, 2005, pp. 133–144.

[10] X. Wu and R. Srikant, "Scheduling efficiency of distributed greedy scheduling algorithms in wireless networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 6, pp. 595–605, Jun. 2007.

[11] C. Joo, X. Lin, and N. B. Shroff, "Understanding the capacity region of the greedy maximal scheduling algorithm in multi-hop wireless networks," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 1103–1111.

[12] P. Chaporkar, K. Kar, X. Luo, and S. Sarkar, "Throughput and fairness guarantees through maximal scheduling in wireless networks," *IEEE Trans. Inf. Theory*, vol. 54, no. 2, pp. 572–594, Feb. 2008.

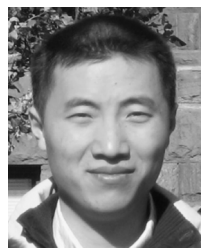
[13] G. Sharma, R. Mazumdar, and N. B. Shroff, "On the complexity of scheduling in wireless networks," in *Proc. ACM MobiCom*, 2006, pp. 227–238.

[14] P. Kyasanur, J. So, C. Chereddi, and N. H. Vaidya, "Multichannel mesh networks: challenges and protocols," *IEEE Commun. Mag.*, vol. 13, no. 2, pp. 30–36, Apr. 2006.
[15] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Found. Trends Netw.*, vol. 1, no. 1, pp. 1–144, 2006.
[16] X. Lin and S. Rasool, "Distributed and provably-efficient algorithms for joint channel-assignment, scheduling and routing in multi-channel ad hoc wireless networks," *IEEE/ACM Trans. Netw.*, vol. 17, no. 6, pp. 1874–1887, Dec. 2009.
[17] S. Merlin, N. H. Vaidya, and M. Zorzi, "Resource allocation in multi-radio multi-channel multi-hop wireless networks," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 610–618.
[18] C. Joo and N. B. Shroff, "Performance of random access scheduling schemes in multi-hop wireless networks," *IEEE/ACM Trans. Netw.*, vol. 17, no. 5, pp. 1481–1493, Oct. 2009.
[19] G. Sharma, C. Joo, and N. B. Shroff, "Joint congestion control and distributed scheduling for throughput guarantees in wireless networks," *Trans. Model. Comput. Simulation*, vol. 21, no. 1, Dec. 2010, Art. no. 5.
[20] L. Ying, S. Shakkottai, and A. Reddy, "On combining shortest-path and back-pressure routing over multihop wireless networks," in *Proc. IEEE INFOCOM*, 2009, pp. 1674–1682.
[21] L. Jiang and J. Walrand, "A distributed CSMA algorithm for throughput and utility maximization in wireless networks," *IEEE/ACM Trans. Netw.*, vol. 18, no. 3, pp. 960–972, Jun. 2010.
[22] P. Gupta and P. R. Kumar, "The cpacity of wireless networks," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 388–404, Mar. 2000.
[23] G. Sharma, C. Joo, and N. B. Shroff, "Distributed scheduling schemes for throughput guarantees in wireless networks," in *Proc. 43rd Annu. Allerton Conf. Commun., Control, Comput.*, Sep. 2006.
[24] V. Bhandari and N. H. Vaidya, "Scheduling in multi-channel wireless networks," in *Proc. Distrib. Comput. Netw.*, 2010, vol. 5935, Lecture Notes in Computer Science, pp. 6–17.
[25] J.-G. Choi and G. S. Choi, "Optimal scheduler with simplified queue structure for mutli-channel wireless networks," *IEEE Commun. Lett.*, vol. 15, no. 9, pp. 962–964, Sep. 2011.
[26] D. Qian, D. Zheng, J. Zhang, and N. Shroff, "CSMA-based distributed scheduling in multi-hop MIMO networks under SINR model," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.
[27] K. Jain, J. Padhye, V. Padmanabhan, and L. Qiu, "Impact of interference on multihop wireless network performance," in *Proc. ACM MobiCom*, 2003, pp. 66–80.
[28] H. Li, Y. Cheng, C. Zhou, and P. Wan, "Multi-dimensional conflict graph based computing for optimal capacity in MR-MC wireless networks," in *Proc. IEEE ICDCS*, Jun. 2010, pp. 774–783.
[29] J. Tang, S. Misra, and G. Xue, "Joint spectrum allocation and scheduling for fair spectrum sharing in cognitive radio wireless networks," *Comput. Netw.*, vol. 52, no. 11, pp. 2148–2158, 2008.
[30] K. N. Ramachandran, E. M. Belding, K. C. Almeroth, and M. M. Buddhikot, "Interference-aware channel assignment in multiradio wireless mesh networks," in *Proc. IEEE INFOCOM*, 2006, pp. 1–12.
[31] Y. Cheng *et al.*, "Virtual network approach to scalable IP service deployment and efficient resource management," *IEEE Commun. Mag.*, vol. 43, no. 10, pp. 76–84, Oct. 2005.
[32] V. A. Siris, E. Z. Targos, and N. E. Petroulakis, "Experiences with a metropolitan multiradio wireless mesh network: design, performance, and application," *IEEE Commun. Mag.*, vol. 50, no. 7, pp. 128–136, Jul. 2012.
[33] S. Avallone and G. D. Stasi, "A new MPLS-based forwarding paradigm for multi-radio wireless mesh networks," *IEEE Trans. Wireless Commun.*, vol. 12, no. 8, pp. 3968–3979, Aug. 2013.

**Yu Cheng** (S'01–M'04–SM'09) received the B.E. and M.E. degrees in electronic engineering from Tsinghua University, Beijing, China, in 1995 and 1998, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2003.

From 2004 to 2006, he was a Postdoctoral Research Fellow with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada. Since 2006, he has been with the Department of Electrical and Computer Engineering, Illinois Institute of Technology, Chicago, IL, USA, where he is now an Associate Professor. His research interests include next-generation Internet architectures and management, wireless network performance analysis, network security, and wireless/wireline interworking.

Dr. Cheng is an Associate Editor for the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY and the New Books and Multimedia Column Editor for *IEEE Network*. He served as a Co-Chair for the Wireless Networking Symposium of IEEE ICC 2009, a Co-Chair for the Communications QoS, Reliability, and Modeling Symposium of IEEE GLOBECOM 2011, a Co-Chair for the Signal Processing for Communications Symposium of IEEE ICC 2012, a Co-Chair for the Ad Hoc and Sensor Networking Symposium of IEEE GLOBECOM 2013, and a Technical Program Committee (TPC) Co-Chair for WASA 2011. He is a founding Vice Chair of the IEEE ComSoc Technical Subcommittee on Green Communications and Computing. He received a Best Paper Award from the conferences QShine 2007 and ICC 2011. He received the National Science Foundation (NSF) CAREER Award in 2011 and the IIT Sigma Xi Research Award in the junior faculty division in 2013.

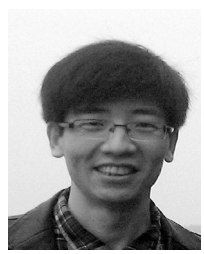


**Hongkun Li** (M'09) received the B.S. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2004, the M.E. degree in electrical engineering from Beijing University of Posts and Telecommunication (BUPT), Beijing, China, in 2007, and the Ph.D. degree in computer engineering, focusing on optimal resource allocation and capacity analysis of multiradio multichannel wireless network, from the Illinois Institute of Technology, Chicago, IL, USA, in 2012.

He joined InterDigital Communications, LLC, King of Prussia, PA, USA, in 2012. His current research focuses on M2M, IoT, and proximity peer-to-peer communication, including technology development and standardization.

**Devu Manikantan Shila** received the M.S and Ph.D. degrees in computer engineering, focused in the area of design and optimization of next generation multihop wireless networks, from the Illinois Institute of Technology, Chicago, IL, USA, in 2007 and 2011, respectively.

As a Sr. Research Scientist with the United Technologies Research Center (UTRC), Hartford, CT, USA, she mainly conducts research in the area of analyzing, designing, and developing secure methodologies for embedded systems. She has published heavily along the lines of interference and secure-aware algorithm design and analysis. She has security expertise in a variety of topics including reverse engineering, modeling risks and vulnerabilities at the physical, system, network and application level, and design and theoretical analysis of secure protocols and algorithms. She has research and development expertise in both the design and analysis of secure and resilient algorithms and protocols for wireless and embedded systems.



**Xianghui Cao** (S'08-M'11) received the B.S. and Ph.D. degrees in control science and engineering from Zhejiang University, Hangzhou, China, in 2006 and 2011, respectively.

During 2007 to 2009, he was a Visiting Scholar with the Department of Computer Science, University of Alabama, Tuscaloosa, AL, USA. His research interests include wireless network performance analysis, energy efficiency of wireless networks, networked estimation and control, and network security.

Dr. Cao is an Associate Editor of the *KSII Transactions on Internet and Information Systems* and *Security and Communication Networks*. He is a TPC member of IEEE GLOBECOM 2013 and 2014, IEEE ICC 2014, IEEE VTC 2013 and 2014, etc.